Description

EMPIRICAL DATABASE ACCESS ADJUSTMENT

Inventors

Harlan Seymour, Sourabh Satish, Anne Yeh, and Benjamin Yeung

Related Applications

Commonly assigned U.S. patent application serial no. 10/612,198 filed July 1, 2003, entitled "Real-Time Training for a Computer Code Intrusion Detection System" and commonly assigned U.S. patent application serial no. 10/632,857 filed July 31, 2003, entitled "Computer Code Intrusion Detection System Based on Acceptable Retrievals" are hereby incorporated by reference in their entireties into the present patent application.

Technical Field

This invention pertains to the field of thwarting intrusions to computer databases perpetrated by malicious attackers.

Background Art

Databases are perhaps the most critical resource of an enterprise, and therefore it is of prime importance to secure them. Standard database products have access control interfaces for setting permissions on the defined tables and columns of the database. Such access control interfaces can allow, deny, or

revoke permissions for a given user and for a given operation on each database table and column.

Databases are managed by database administrators (DBAs), who determine access control settings for the database by participating in the design and implementation of the applications using the database and/or by reviewing suggestions made by the application developer or provider. The DBA obtains knowledge of users, groups, roles, and applications accessing the database. This information is not always readily and easily available.

In any case, the access control settings are configured on the basis of perceived application behavior, i.e., the application is expected to access specific parts of the database for specific operations, and as such, the DBA chooses to apply relevant access control settings. This process is complicated, since database applications are typically huge, with many components developed by large teams. Thus, to consolidate all the features to focus access to parts of the database for specific operations is almost impossible. As a result, it is often the case that unnecessarily loose (open) access control settings are applied to the database to account for the various unknowns.

These loose access control settings give rise to concerns that malicious individuals are thereby able to access parts of

the database that could have been protected without disturbing access by benign individuals (or applications). They are not prevented from doing so by conventional techniques, since said access is within the bounds of permissible access settings as originally configured by the DBA.

What is needed is a solution for ensuring that minimal access control settings are applied to the database, so that each application can continue to function as usual, while avoiding loose settings.

Disclosure of Invention

Computer implemented methods, apparati, and computer-readable media for empirically adjusting access to a database (1). An apparatus embodiment comprises: coupled to the database (1), a database discovery module (11) for determining authorized accesses to the database (1); coupled to the database (1), a command monitoring module (12) for monitoring actual accesses to the database (1); and coupled to the database discovery module (11) and to the command monitoring module (12), an analysis module (13) for comparing actual accesses with authorized accesses.

Brief Description of the Drawings

These and other more detailed and specific objects and features of the present invention are more fully disclosed in the

following specification, reference being had to the accompanying drawings, in which:

Figure 1 is a block diagram illustrating apparatus modules suitable for use in the present invention.

Figure 2 is a flow diagram illustrating a method embodiment of the present invention.

Detailed Description of the Preferred Embodiments

"Database" is used broadly herein to comprise any collection of data stored on any computer readable medium. A database normally comprises tables and columns, and is accessed by some query language such as SQL (Structured Query Language).

"Coupled" is used broadly herein to encompass any type of direct or indirect communicative coupling.

Figure 1 illustrates apparatus suitable for carrying out the present invention. Database 1 can be any type of database, such as a relational database or a flat file. When database 1 is a relational database, commands 15 are typically written in a SQL language. As used herein, "SQL" is taken in the broad sense to mean the original language known as SQL (Structured Query Language, which originated in the IBM Research Labs in 1969), any derivative thereof, or any structured query language used for accessing a relational database.

SQL 92 is the current standard version of SQL. It is published in many places, including on the World Wide Web. Each

vendor of a database 1 tends to have its own flavor of SQL, a flavor compatible with SQL 92.

An example of a SQL command 15 is the SELECT command:

SELECT C1,C2 FROM T1,T2

In the above command, a user is attempting to select columns C1 and C2 from T1 and T2. Each of T1 and T2 can be a table or a view. (Views are discussed below.) Without additional information, we don't know whether C1 belongs to T1 or T2; and we don't know whether C2 belongs to T1 or T2.

GRANT, DENY, and REVOKE commands can be used to limit access to database 1. Not every database vendor has a REVOKE command. For example, Oracle does not. In this case, DENY is used instead. Most access control modules 16 work on the basis that a DENY command overrides a GRANT command.

In the case where database 1 is not a relational database, the commands can be written in a language other than SQL, such as XML.

Database 1 may have associated therewith an internal audit table 10 and/or an external database log file 19 for storing audit and/or ancillary information pertaining to database 1. Database 1 is typically packaged within a dedicated computer known as a database server, which may also contain communications and other modules. The database server can contain more than one database 1.

Access control module 16 is a hardware, firmware, or software device that determines what users and operations can access the various tables and columns within database 1. Access control settings within module 16 can be set and/or changed by a human database administrator (DBA) 17.

Database discovery module 11 and command monitoring module 12 are coupled to database 1. The purpose of database discovery module 11 is to i.) determine database 1 structure in terms of its tables and columns, and in terms of artifices (views, stored procedures, etc.) that manipulate the tables and columns, and ii.) determine (uncover) authorized (permitted) accesses to database 1. The purpose of command monitoring module 12 is to monitor actual accesses to database 1. Analysis module 13 is coupled to modules 11 and 12. The purpose of analysis module 13 is to compare actual database accesses with authorized accesses, and to make and implement appropriate decisions based upon the results of such comparisons. Module 13 can generate one or more third party reports 18. Storage area 14 is coupled to modules 12 and 13. The purpose of storage area 14 is to accumulate data generated by command monitoring module 12 during a training phase (observing step 22). Storage area 14 may be part of monitoring module 12.

Modules 11-14 can be implemented in hardware, firmware, software, or any combination thereof. When implemented in

software, modules 11-14 can reside on one or more computer-readable media, such as one or more hard disks, floppy disks, CDs, DVDs, etc.

Command monitoring module 12 is usually a sniffer, because a sniffer does not modify the input stream of data emanating from commands 15. Alternatively, monitoring module 12 can be a proxy. A proxy does affect the input stream, because the queries it receives must be rerouted to the database server. The inputs to module 12 are a plurality of commands 15. Figure 1 illustrates n such commands, where n is a positive integer. Module 12 examines the data streams generated by commands 15 and extracts the relevant information therefrom. Module 12 decrypts commands 15 if commands 15 are encrypted.

A method embodiment of the present invention will now be described in conjunction with Figure 2.

At step 21, authorized accesses of database 1 are discovered by database discovery module 11. An "authorized access" comprises the following combination: the authorized database 1 (in those embodiments where the database server has more than one database), the authorized table, the authorized column, the authorized operation, and the authorized user. An individual record produced by database discovery module 11 can have the forms:

Permitted[database][table][operation][user]=0 or 1

Permitted[database][table][column][operation][user]=0 or 1

INSERT and DELETE permission settings are made only at the table level, not the column level. One cannot "insert" or "delete" an individual column as this is really an UPDATE operation. So, we will be granting/denying INSERT and DELETE permissions only at the table level. SELECT and UPDATE permission settings can be made at a column granularity, or at a table granularity. Typically, we will be granting/denying SELECT and UPDATE at the column level on a per user basis, unless all of the columns of a given table are granted/denied for a given user for a SELECT or UPDATE, in which case we will do the granting/denying at the table level.

A view V is a command 15 involving multiple tables. An example of a view V is:

CREATE V AS SELECT C1,C2 FROM T1,T2

The discovery step 21 ascertains the definitions of the views associated with database 1, i.e., discovery module 11 resolves the views into individual accessed tables and columns along with the operations used on them. Thus, for example, if a command 15 monitored during the observing step 22 is "SELECT A from V", it is known what columns and tables within database 1 are implicated because discovery module 11 has determined this information.

A stored procedure is a procedure that is performed on the data within database 1. An example of a stored procedure SP1 is:

CREATE SP1 (X,Y) AS STORED PROCEDURE

In the above example, X and Y are parameters representing users, commands, tables, columns, dates, functions, constants, etc. Again, database discovery module 11 resolves (breaks down) each stored procedure into accessed tables and columns along with the operations used on them.

The database discovery module 11 must resolve (break down) any database 1 artifice that performs operations on database tables or columns in a "black box" manner, as do views, stored procedures, user-defined functions, triggers, etc.

In addition to views and stored procedures, user-defined functions and triggers will also be "discovered". In fact, any SQL entity that reads/writes column values or table rows must be discovered. For example, min/max column value checking can be specified using values from other columns. Also, there can be many levels of nesting. T1 (from the above SELECT example) can really be a view (V). But V itself might be composed of other views (and tables), ad infinitum.

Module 11 can be programmed to automatically determine the relevant information. Alternatively, or in addition to information determined automatically by module 11, a human such as DBA 17 can provide information to analysis module 13 regarding

authorized combinations. This can be done, for example, if there are complicated views, stored procedures, etc., associated with database 1.

The information uncovered by database discovery module 11 is stored in a storage area, for later use by command monitoring module 12. The storage area can be part of database discovery module 11, or a separate module.

During the observing step (training phase) 22, command monitoring module 12 monitors incoming commands 15, keeping track (by means of updating storage area 14) as to which users perform which operations on which tables and columns within database 1. There are four main operations that are tracked. SELECT is used to read a column value. UPDATE is used to modify (write) a column value. INSERT is used to insert a new row of column values into a table. DELETE is used to delete an existing row of column values from a table.

Monitoring module 12 first breaks down commands 15 into their constituent elements using information provided by database discovery module 11. For example, in the illustration given above, the command SELECT C1,C2 from T1,T2 is ambiguous in that it is not known a priori whether C1 belongs to T1 or T2, it is not known whether C2 belongs to T1 or T2, and it is not known whether T1 and T2 are tables or views. But this information is determined by database discovery module 11 in step 21, and

command monitoring module 12 uses this information in step 22 to resolve the command and then store the observed combination in storage area 14. An entry in storage area 14 may have one of the forms:

Observed[database][table][operation][user]= 0 or 1

Observed[database][table][column][operation][user]=0 or 1

The duration of the observing step 22 is normally defined in terms of a preselected time period. Alternatively, the duration of observing step 21 can be defined in terms of a preselected number of entries made to storage area 14. In either case, the duration of the observing step 22 should be sufficiently long that monitoring module resolves and records many commands 15, and all expected functionalities of the applications accessing database 1 are exercised. This will enable analysis module 13 to completely understand database access patterns.

Monitoring module 12 can employ any technique of in-line interception or real-time auditing to obtain the desired information.

Real-time auditing can be used in cases where database 1 has an auditing feature. The auditing information may be placed into an audit table 10 internal to database 1 or into an external database log file 19. In real-time auditing, module 12 instructs database 1 to generate a stream of events every time a command 15 enters database 1. The stream can include such items as the text

of the command 15, a date/time stamp, information pertaining to the user that issued the command 15, the IP (Internet Protocol) address of the issuing computer, the application that issued the command 15, etc. The stream can appear to module 12 in string or binary form, and can be extracted using a number of different techniques, depending upon the implementation, including APIs (Application Programming Interfaces) that access database 1. One example is to use ODBC (Open DataBase Connectivity), a set of C language API's that allows one to examine or modify data within database 1. If the Java programming language is used, JDBC (Java DataBase Connectivity) can be used instead.

Another way that module 12 extracts the needed information from database 1 is to use code injection or patching to inject logic into one or more modules associated with database 1, to transfer control to module 12.

In another embodiment, called "direct database integration", the database 1 vendor, who has access to the commands 15 in conjunction with the normal operation of the database 1, makes the commands 15 available to module 12.

In yet another embodiment, in cases where database 1 supports it, external database log file 19 may be examined without the need to resort to special software.

Once a command 15 has been processed by module 12, the command 15 can optionally be expunged from any table 10 or log file 19 it is stored in, to make room for subsequent commands 15.

Module 12 normally obtains its information in real time, but, alternatively, the information could be extracted in a non-real time manner, e.g., in those embodiments where audit table 10 or log file 19 is used.

At step 23, analysis module 13 compares the actual accesses of database 1, as gathered in storage area 14, with the normally larger universe of authorized accesses as determined by database discovery module 11. Analysis module 13 can generate a map of which parts (tables and columns) of which database 1 were accessed during step 22 via which operations emanating from which users. The map can then be displayed to DBA 17 by any conventional means. For example, the map can be displayed on a computer monitor, with actual accesses being portrayed in one color, and authorized accesses that were not observed during the observing step 22 portrayed in a different color.

At step 24, analysis module 13 and/or DBA 17 adjust access control settings to database 1, based upon results of comparing step 23 and possibly based upon pre-established criteria. The adjustments are made by changing settings within access control module 16. Such adjustments can include one or more of the following:

1) DBA 17 receives a report from module 13 containing suggested revised access control settings that should be applied to database 1. The suggestion is normally to harden database 1, i.e., impose stricter access control settings. For example, the suggestion may be to deny access to operations by certain users on database 1 columns and tables that were preconfigured to be authorized, but which were not observed during observing step 22.

2) Analysis module 13 is allowed to automatically harden database 1 (for all times of the day), i.e., apply access control settings as determined by module 13. Again, the criterion can be to deny access to operations by certain users on certain columns and tables of certain databases that were preconfigured to be authorized, but which were not observed during observing step 22.

3) Module 13 is allowed to harden database 1 dynamically based upon time of day, i.e., access control module 16 is programmed to harden database 1 during certain times of the day but not during other times.

4) DBA 17 can be alerted by module 13 regardless of time of day or in a time-based access pattern. The alerts can convey those combinations of databases, tables, columns, operations, and users that were preconfigured to be authorized, but which were not observed during observing step 22.

5) Command monitoring module 12 can be allowed to continue monitoring commands 15 past the duration of the observing step

22. During the extended time period, for example, analysis module 13 can be programmed to alert DBA 17 in real time regarding commands 15 that are observed by monitoring module 12 during the extended time period but that were not observed during observing step 22.

Adjustments involving the automatic hardening of database 1 can be implemented by using a bypass connection 20 between analysis module 13 and access control module 16 to bypass DBA 17. The hardening commands can be written as standard SQL commands as supported by database 1, or by using database specific proprietary APIs (Application Programming Interfaces). Examples of such APIs are OCI for an Oracle database 1, or DMO for a Microsoft SQL Server. OCI and DMO are libraries that enable programming languages other than SQL to access database 1, by translating the commands into SQL. OCI is a set of subroutines in programming language C. DMO uses COM objects in C or in Visual Basic.

In optional step 25, analysis module 13 generates one or more third party reports 18, e.g., a regulatory compliance report such as the data security report required by HIPAA (Health Insurance Portability and Accountability Act). As used herein, "third party" means a report to be processed by an entity other than DBA 17 or access control module 16.

Normal operations of database 1 can continue uninterrupted during steps 21-25 of the present invention. This is an important attribute.

Returning to Figure 1, we see an exemplary database 1 having one table (People) with four columns (ID, Name, Phone, and Social Security Number). Command monitoring module 12 monitors two incoming commands 15 from user John. Command 15(1) is an UPDATE, and command 15(2) is a SELECT. Module 12 monitors these commands 15 and dissects which operations are being performed on which tables and columns within database 1. At step 23, analysis module 13 concludes that user John has not selected column SSN, has not updated column ID, and has not updated column SSN. (John has implicitly selected column ID, because of the WHERE subcommand embedded within command 15(1)). At step 24, module 13 can perform one or more tasks as described above, such as to automatically deny future access to John to these unobserved combinations, generate an alert to DBA 17 if a subsequent command 15 arrives that attempts to access one of these previously unobserved combinations, etc.

The above description is included to illustrate the operation of the preferred embodiments and is not meant to limit the scope of the invention. The scope of the invention is to be limited only by the following claims. From the above discussion, many variations will be apparent to one skilled in the art that

would yet be encompassed by the spirit and scope of the present invention.

What is claimed is: